INTELLIGENT TUTORING SYSTEMS:
MEASURING STUDENT EFFORT DURING ASSESSMENT

A Thesis
Submitted to the Faculty of Graduate Studies and Research
In Partial Fulfilment of the Requirements
For the Degree of
Masters of Science
in
Computer Science
University of Regina

By
Peter Lach
Regina, Saskatchewan
August, 2013

Copyright 2013: Peter Lach
Peter Lach, candidate for the degree of Master of Science in Computer Science, has presented a thesis titled, *Intelligent Tutoring Systems Measuring Student Effort During Assessment*, in an oral examination held on July 12, 2013. The following committee members have found the thesis acceptable in form and content, and that the candidate demonstrated satisfactory knowledge of the subject material.

External Examiner: Dr. Warren Wessel, Faculty of Education

Co-Supervisor: Dr. Cortney J. Butz, Department of Computer Science

Co-Supervisor: Dr. Robert B Maguire, Department of Computer Science

Committee Member: Dr. Daryl Hepting, Department of Computer Science

Chair of Defense: Dr. Nader Mobed, Faculty of Science
ABSTRACT

Each tutoring system must face the problem of dealing with uncertainty involved in student interactions. The quality of a correct answer to a question may be different for each individual student. Some answers are produced by careful thinking and analysis using memorised facts, where other answers are just guesses. Therefore there is a need to develop a tutoring system which takes into consideration the quality of an answer to produce a more detailed student model. Many tutoring systems developed to this point use question-answer techniques to estimate the student knowledge state, however none of them take into consideration the quality of student answers to estimate a student’s knowledge state. The tutoring system presented here addresses this problem by measuring student effort during assessment.

In this thesis, I present an exploratory study of eye tracking technology which is used to modify the student model in Intelligent Tutoring Systems during assessment to create a more accurate estimate of the student’s knowledge state. This tutoring system uses a Bayesian Network (BN), a formal framework for uncertainty management in Artificial Intelligence based on probability theory, to model the student’s knowledge state. An extra communication channel between the student and the tutoring system is an eye tracking device which is used to obtain real-time data about the student’s eye activities. Using this additional data about the student allows the ITS to create a more precise model of the student knowledge state, which at the end leads to better adaptation of the learning instructions. In addition, I describe the architecture of
this tutoring system and the role of each component in the system.

The tutoring system in this thesis sets an example and offers a reference on the application of eye tracking technology in the development of intelligent tutoring systems. The tutoring system presented in this thesis can be useful to any work involving an eye tracking technology due to its ability to extract an eye movement in real time. The implementation of this tutoring system can be used as an example of communication between a tutoring system and an eye tracking device.
ACKNOWLEDGEMENT

I am greatly thankful to my co-supervisors Dr. Brien Maguire and Dr. Cory Butz, who provided me with a motivating, enthusiastic, and critical research atmosphere. Without their guidance, my future research achievements would not have been possible. It is a great pleasure for me to conduct this thesis under their supervision. I thank them for their patience and encouragement that carried me on through difficult times, and for their insights and suggestions that helped to shape my research skills. Their valuable feedback contributed greatly to this thesis.

I am thankful to the Faculty of Graduate Studies and Research, the Department of Computer Science, Dr. Brien Maguire and Dr. Cory Butz for the financial support and computer facilities provided, which allowed me to concentrate on my research and complete my thesis.

I also would like to thank to Dr. Daryl H. Hepting for allowing me to work in Regina Interactive Cognitive Experimentation Lab.
DEDICATION

I dedicate this thesis to my loving wife Batgerel Jambaldorj. Without her encouragement and help, I could not maintain a high level of confidence and enthusiasm in my research. When I need her most, she is always there and has supported me through all these years.
Post defense acknowledgment

I am thankful to my external examiner Dr. Warren Wessel and chair Dr. Nader Mobed. Their valuable feedback helped me to improve the thesis in many ways.
# TABLE OF CONTENTS

ABSTRACT i

ACKNOWLEDGEMENT iii

DEDICATION iv

POST DEFENSE ACKNOWLEDGEMENT v

TABLE OF CONTENTS vi

LIST OF FIGURES x

LIST OF ALGORITHMS xi

1 INTRODUCTION 1

1.1 Problem Statement 2

1.2 Thesis Contribution 4

1.3 Thesis Layout 5

2 BACKGROUND KNOWLEDGE 6

2.1 Intelligent Tutoring Systems 6

2.1.1 Domain Knowledge Model 7

2.1.2 Student Model 10

2.1.3 Tutoring Model 10
2.2 A closer look at the student model ........................................ 12
  2.2.1 Uncertainty Management ............................................. 13
  2.2.2 Bayesian Networks .................................................... 14
2.3 Eye Tracking ............................................................ 16
2.4 Related Work ........................................................... 18
2.5 Summary ................................................................. 19

3 MENTAL WORKLOAD AS MEASURE OF EFFORT .................. 21
3.1 Mental Workload ......................................................... 21
3.2 Measuring Mental Workload ............................................. 24
  3.2.1 Index of Cognitive Activity ......................................... 25
  3.2.2 Wavelet Analysis ...................................................... 26
3.3 Summary ................................................................. 29

4 ARCHITECTURE OF PROPOSED SYSTEM ......................... 31
4.1 Knowledge Domain Module ............................................ 32
  4.1.1 Course Teaching Material .......................................... 32
  4.1.2 Assessment Question Set and Solution Keys ................... 34
4.2 Student Module ......................................................... 38
  4.2.1 Building student model using a Bayesian Network ............ 38
  4.2.2 Updating student knowledge state ............................... 43
4.3 Eye Tracking Device ..................................................... 43
4.4 User Interface Module .................................................. 45
4.5 Tutoring Module ........................................................ 49
4.6 Adaptive Learning Environment ...................................... 49
  4.6.1 Best Topic To Learn ................................................ 49
  4.6.2 Concept Prerequisites ................................................ 50
  4.6.3 Learning Path Generator .......................................... 51
# LIST OF FIGURES

2.1 Intelligent Tutoring Systems Component Architecture ............................ 7
2.2 A BN on variables $U = \{A, B, C, D, E, F\} .................................. 15$
3.1 Radial and Circular Muscles ................................................................. 22
3.2 Raw Pupil Dilation Signal during calibration step ................................. 28
3.3 Pupil Dilation Signal after two step wavelet transformation ................. 29
4.1 System Architecture ................................................................. 32
4.2 Data structure tree split ................................................................. 33
4.3 Knowledge Domain Tree ................................................................. 34
4.4 Sample Question to Assess Student Knowledge. ................................. 35
4.5 Sample Question Including a Graphical Aid. .................................... 35
4.6 Assessment Question Tree ................................................................. 37
4.7 Bayesian network model for queue data structure concepts ................. 39
4.8 Bayesian Network Student Model ....................................................... 41
4.9 Bayesian Network implemented as 2D array .................................... 42
4.10 Typical EyeLink II System Configuration .......................................... 44
4.11 Correct Camera Setup ................................................................. 45
4.12 Progress Screen ................................................................. 47
4.13 Concept Screen ................................................................. 48
4.14 Next Best Topic To Learn ................................................................. 50
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.15</td>
<td>Removing an element from Queue with all pre-requisites</td>
<td>51</td>
</tr>
<tr>
<td>4.16</td>
<td>Bayesian Network for assessing student knowledge state</td>
<td>58</td>
</tr>
<tr>
<td>A.1</td>
<td>Access to latest data produced by the eye tracking device</td>
<td>79</td>
</tr>
<tr>
<td>A.2</td>
<td>Method which updates each word as read</td>
<td>80</td>
</tr>
<tr>
<td>A.3</td>
<td>Method which creates a list of pre-requisites</td>
<td>80</td>
</tr>
<tr>
<td>A.4</td>
<td>Signal Transform Method</td>
<td>81</td>
</tr>
<tr>
<td>A.5</td>
<td>Method to open and setup connection to the eye tracker</td>
<td>82</td>
</tr>
</tbody>
</table>
List of Algorithms

1. Algorithm which generates position coordinates for each word. . . . . . 54
2. Algorithm for computing ICA. . . . . . . . . . . . . . . . . . . . . . . . . . 57
Chapter 1

INTRODUCTION

All Intelligent Tutoring Systems (ITSs) share the same goal: to provide tutorial services that support learning. ITSs are computer systems that can be used in learning and contain intelligence [37, 44]. The first generation of computer systems used for education emerged in the 1970s as Computer Assisted Instruction (CAI). CAI was a mature and promising technology with a targeted market [23, 47]. The educational system was looking for solutions to overcome its limitations in dealing with large groups in schools. However, these early systems had significant limitations. They did not consider the diversity of student knowledge levels and did not provide an adaptive learning environment. Education is one of the most demanding fields to which Artificial Intelligence (AI) might be applied. In 1984 [5], Bloom published an article demonstrating that one-on-one tutoring is twice as effective as group teaching. AI researchers saw a solid foundation upon which they could create intelligent systems that would provide effective tutoring for every student. ITSs are intended to take on some of the work of a human teacher or tutor, and to take at least some of the initiative in an educational dialogue. The design and development of such tutors lie at the intersection of computer science, cognitive psychology and educational research; this intersecting area is normally referred to as cognitive science. Human teachers
learn about students through years of experience. Master teachers often use secondary learning features, (e.g., a student’s facial expression, body language, and tone of voice) to augment their understanding of a student’s learning [37]. Interaction between students and teachers provides critical data about a student’s goals, skills, motivation, and interest. For the teacher to adapt teaching instruction to a student’s needs, the teacher needs to first observe the student’s behaviour. The more the teacher can observe, the better he/she can understand what the student may need in a given moment. More details about the student can lead to more effective adaptation of teaching instruction. ITSs can adapt instruction to a student’s particular knowledge state. They interact with the student to determine the student’s knowledge state. A greater level of adaptation is required to build more accurate student models which are based on more information being available during interaction. Data gathered in student models often includes uncertainty. One formal framework in artificial intelligence to deal with uncertainty is Bayesian Network (BN) [39]. BNs are used in ITSs to support classification and prediction, model student knowledge, predict student behaviour, make tutoring decisions, and determine on which steps the student will need help and the student’s probable method for solving problems [34]. One of the problems affecting the learning process in Intelligent Tutoring Systems is for the tutoring system to accurately estimate the student’s knowledge state. In this thesis, I present an exploratory study of eye tracking technology which is used to modify the student model in ITSs during assessment to create a more accurate estimate of the student’s knowledge state.

1.1 Problem Statement

No intelligent communication can take place without a certain understanding of the recipient [53]. Student diagnosis is the most difficult tutoring task. It requires long
term basic research before a solution can be found [54].

Human teachers are capable of adapting the teaching instruction based on what the students need at a given moment. By observing the student behaviour, a teacher can identify if the student has a problem in understanding a certain concept. It is important for the teacher to have a good understanding of what the student needs because the more the teacher knows about the student the better the teacher can adapt the teaching instructions, which leads to more effective learning. This ability of a human teacher is therefore required to be reproduced in the ITs. The intelligent tutors make inferences about presumed student knowledge and store them in the student model [37].

The student model is considered to be the central module because it stores information which is specific to each individual student. Crafting such a model involves techniques to represent content skills, knowledge about learning, and affective characteristics. The fundamental reason why a student model is included in intelligent tutoring systems is to ensure that the system has adequate knowledge about each student so it can respond effectively, engage student interest and promote learning [37]. Without an explicit student model, the tutoring system is unable to make decisions regarding adaptation of instructional content and guidance, and is forced to treat all students identically. Due to computational complexity, we cannot capture all aspects of student behaviour. For example, it is currently impossible to capture the mood of the student during learning. Therefore student models are only partial models, and provide only an approximate representation of the student. An additional drawback for computers is that their communication channel is very limited, typically to a keyboard, mouse, and screen. The communication channel between human teachers and the students is much richer because the human teachers use their full sensory system to observe a student’s behaviour. Each tutoring system must face the problem of dealing with uncertainty involved in student interactions. Not all correct answers to
the same question are identical. Some answers are produced by careful thinking and analysis using memorised facts, where other answers are just guesses. Therefore there is a need to develop a tutoring system which also takes into consideration the quality of an answer to produce a more detailed student model. Answering a question requires a student to apply learned knowledge in practice, or recall facts from memory. These processes have an effect on cognitive workload. This workload can be observed and measured using eye tracking technology.

1.2 Thesis Contribution

This thesis proposes a new intelligent tutoring system using eye tracking technology for students to learn about data structures. This system uses Bayesian Network to model student knowledge state, which is a formal framework for uncertainty management in Artificial Intelligence based on probability theory. The extra communication channel between the student and the tutoring system is the eye tracking device which is used to obtain real-time data about the student’s eye activities. Beside the response from the student to a question, the tutoring system uses input from the eye tracking device to create a model of the student’s reading pattern, and to assess mental workload during question answers. This thesis introduces this new technique which improves the model and provides more precise assessment of the student’s attention. Using the distance formula we are able to compute precisely which words on the screen were read by the student. In this thesis, I introduce a method of measuring effort while answering a question. Using this method, the tutoring system has the ability to determine if there is an increased mental workload and use this new information to update the student knowledge model state. Using additional data about the student we can create a more precise model of the student’s knowledge state, which in the end leads to better adaptation of the learning instructions. In this manner, we can address
the problem of a tutoring system dealing with uncertainty during student interaction. Better adapted instructions support the student’s learning activities. I discuss how to use the Bayesian Network to guide the student’s learning processes. Later, I describe in detail the system architecture, the system’s adaptation abilities and the effort assessment module which provides more precise assessment of the student knowledge state.

1.3 Thesis Layout

This thesis is organised into five chapters. Chapter 2 reviews Intelligent Tutoring Systems technology research and eye tracking technology. In Chapter 3, I discuss pupillary response and its correlation to mental workload. This chapter also introduces the Index of Cognitive Activity as a framework for measuring mental workload. In Chapter 4, I describe the architecture of Intelligent Tutoring Systems Measuring Student Effort During Assessment and the role of each component in the system. The conclusions and future work are presented in Chapter 5.
Chapter 2

BACKGROUND KNOWLEDGE

In this chapter I review what Intelligent Tutoring Systems are, how they are structured from an architectural point of view, and discuss problems which need to be considered during development of a tutoring system. Furthermore, I describe the student module in more detail, its importance in one-on-one tutoring, and how we can use a Bayesian Network to deal with uncertainty in the student’s knowledge state. In addition, I briefly introduce eye tracking technology. At the end of this chapter, I review the current research in Intelligent Tutoring Systems involving the use of eye tracking technology to improve the learning experience.

2.1 Intelligent Tutoring Systems

All ITSs share the same goal: to provide tutorial services that support learning. ITSs are computer systems that can be used in learning and that contain intelligence. The first generation of computer systems used for education emerged in the 1970s as Computer Assisted Instruction (CAI) software. CAI was a promising technology with a targeted market. The educational system was looking for solutions to deal with large student numbers in schools. However, these early systems had their own limitations; they did not consider the diversity of student knowledge levels and did
not provide adaptive learning environments. Education is one of the most demanding fields in which AI is applied. In 1984 [5], Bloom published an article demonstrating that one-on-one tutoring is twice as effective as group teaching. AI researchers saw a solid foundation upon which they could create intelligent systems that would provide effective tutoring for every student. ITSs are intended to take some of the role of a human teacher or tutor, and to take at least some of the initiative in an educational dialogue. ITSs are typically separated into several different parts, where each part plays an individual role. The basic four-component architecture of ITSs consists of the domain model, the student model, the tutoring model and the user interface component [44]. Figure 2.1 shows how these four components interact with each other.

![Figure 2.1: Intelligent Tutoring Systems Component Architecture](image)

### 2.1.1 Domain Knowledge Model

The Domain Knowledge Model, also known as expert knowledge, contains the concepts, the rules and the problem-solving strategies of the domain to be learned. This module can fulfill multiple roles: as a source of knowledge being taught, or as an assessment module of the student’s knowledge state. The development and implementation of a domain knowledge model is a difficult problem which has been the subject of numerous AI research projects. Describing knowledge is a very complex task. Knowledge representation in our heads is different from the knowledge repre-
sentation in our computers. Each tutoring system must include a domain-specific expert module that is able to generate and solve domain problems and also perform sound reasoning using this domain knowledge.

When developing an ITS, the epistemological perspective needs to be considered. During the epistemological analysis we define the ontology and inference model of a knowledge-based system. Ontology represents the conceptual model of domain knowledge which focuses on the nature, the properties and the constraints that govern the existence of knowledge. The inference model is the conceptual representation of the nature of the inference structure required to solve a problem, or to execute a task by managing the ontology [37]. An important part of epistemology is to define the nature of the knowledge which we are trying to teach.

With ITSs in mind, there are three known types of domain module models: black box models, glass box models and cognitive models. The black box model is one that generates the correct input-output behaviour over a range of tasks in the domain. The internal computations producing its behaviour are either not available, or are of no use in delivering instruction [1]. An example of a black box model is the system called SOPHIE I [6]. This tutoring system uses an expert system to evaluate the measurements made by students when troubleshooting electronic circuits. The system works mainly through question answering and hypothesis evaluation. SOPHIE I can demonstrate a wide variety of procedures, but it cannot explain what it is doing in the context of an "understanding" of the student.

A glass box model is an intermediate model that reasons in terms of the same domain as the human expert. Developing this module requires knowledge engineering and domain expertise. Glass-box models simulate the human thought processes used in the task being instructed, providing an accurate diagnosis of students’ knowledge and misconceptions. But these models are very expensive to build and the feasibility of their achievement is not proved in many domains [54]. An example of a system
using the glass box model is GUIDON [37]. GUIDON is a tutoring system built on top of an expert system called MYCIN [47]. The basic instruction in GUIDON is driven by T-rules. T-rules are defined on the differential between the expert’s behaviour and the student’s behaviour, but they are also defined on the expert’s reasoning processes [1].

The third model, the cognitive model, aims to match representation formalism and inference mechanism with human cognition by developing a simulation of human problem solving in a domain in which the knowledge is decomposed into meaningful, humanlike components and deployed in a human-like manner. Systems built with the knowledge domain module using the cognitive model are using a cognitive architecture such as the Adaptive Control of Thought (ACT-R). According to ACT, all knowledge begins as declarative information; procedural knowledge is learned by making inferences from already existing factual knowledge. ACT supports three fundamental types of learning: generalization, in which productions become broader in their range of application; discrimination, in which productions become narrower in their range of application; and strengthening, in which some productions are applied more often [2].

Representing knowledge in the knowledge domain is done using carefully selected representation language. The selection should take four important points into consideration: the expressivity of the language, the inference capacity of the language, the cognitive plausibility of the language and the pedagogical orientation of the language. There are several types of representation languages, where each type is suited to different forms of domain knowledge. These languages are typically based on logic and mathematics and have easily parsed grammars and well-defined semantics to ease machine processing. Between the common general-purpose knowledge representation languages we can find production rules, semantic networks, conceptual graphs and frame-based representation.
2.1.2 Student Model

This module is considered to be the central module in ITS because it stores information which is specific to each individual student. The student model is viewed as a dynamic model that implements several functions [37]. According to Wenger [53] the student model must:

1. gather implicit and explicit data about the student,
2. use this data to create a representation of the student’s knowledge state,
3. be able to assess the student’s knowledge level by comparing the student’s knowledge state to expert knowledge state.

A student model in an intelligent tutor observes student behaviour and creates a qualitative representation of the student’s cognitive and affective knowledge. Section 2.2 discusses the student module in more detail including how to manage uncertainty using BN.

2.1.3 Tutoring Model

The tutoring model represents the teacher and takes input from the domain and student models. It makes decisions about tutoring strategies and actions. Modelling the tutoring module is a difficult task because tutoring functions are neither uniformly represented, nor consistently distinct from other functions or components in the system. Several questions need to be answered before designing a tutoring model. Are the functions of the tutoring module distributed throughout the system? Are the tutoring functions integrated with the student model? Should the tutoring model be represented as one component only? Answers to these fundamental questions govern the architectural design of the tutoring model.
Tutoring with ITSs involves guidance from the tutor, and interaction between the tutor and the learner. The fundamental feature of tutoring is interaction between the tutor and the student and ongoing adaptation of the tutor to the student [37]. The main challenge here is to incorporate interactions which can create precise adaptation and tutoring behaviour using real-time data from regular interaction between the learner and the system. Interactivity is important in ITSs because according to VanLehn [51], the more interactive the tutoring, the larger the learning gains that are produced. Therefore one-on-one tutoring should be the most effective. The key to success of tutoring in ITSs is to design effective tutorial interaction.

There are several different designs which have been used in existing ITSs. Among them the most common is Socratic dialogue which enables the most natural form of interaction. An alternative to tutoring through dialogue is tutoring through actions by the use of a rich interface, such as in Andes [37]. Andes is a tutoring system designed to teach introductory physics. This system uses popup messages to inform the student about errors. The student can access more information about the error by using the help buttons. If students are not sure what to do next they can access hint messages. Adding animated agents which are in charge of tutoring is another alternative [25].

Other functions of the tutoring module are to assist and assess the student during learning. The tutoring model makes decisions about when and how to assist the learner. Tutoring systems based on the theory of ACT-R (which are also prime examples of cognitive tutors), make distinctions between declarative and procedural knowledge. These systems use model-tracing algorithms which trace a student’s cognitive steps by parallel application of a series of production rules to facts relating to the problem to be solved. As a student works his way through the problems, his mastery of each rule is inferred by another algorithm, that traces knowledge. The knowledge tracing algorithm uses the student model and the student’s last input to
find production rules which can lead to the current student’s input. The set of production rules are if-then statements composed of actions to be taken when certain conditions are met. In knowledge tracing, each rule is assigned a probability of being known by the student. Rules that are correctly used have their probability adjusted through a Bayesian learning formula. The tutor then uses the resulting estimated level of mastery for each rule to select which problem should be next presented to the student [37].

2.2 A closer look at the student model

The student model in intelligent tutoring systems represents the student’s general knowledge (domain being taught), learning styles, attitudes, and emotions. Crafting such a model involves techniques to represent content skills, knowledge about learning, and affective characteristics. The fundamental reason for including a student model in intelligent tutoring systems is to ensure that the system has adequate knowledge about each student so it can respond effectively, engage student interest and promote learning [37]. Using the student model, the system is able to adapt instructions which match the student’s learning style, which may increase the student’s interest in learning. Due to computational complexity, we can not capture all aspects of student behaviour. Therefore student models are only a partial or an approximate representation of the student knowledge. When building a student model, we need to consider two issues: representing and updating student knowledge.

Representing student knowledge can have many forms, from simple numeric rankings about student mastery to complex plans or networks explaining student knowledge [7] [17]. We use methodology from artificial intelligence (knowledge representation) to encode concepts in a computer in a manner that provides efficient operations on these concepts so computers can reason about concepts and begin to appear intel-
Student models also represent affective characteristics e.g., student emotions and attitudes. Confusion, frustration, excitement, boredom, as well as motivation, self-confidence, and fatigue have been represented. Updating student knowledge is the second issue. Updating is used to infer a student’s current knowledge. Student knowledge improves over several sessions until it reaches the equivalent of the expert in knowledge of the domain. Because of the multiple ways which exist to represent student knowledge, there exist many different techniques which are used to update student knowledge. Cognitive science techniques include model-tracing and constraint-based methods. Artificial intelligence techniques include formal logic, expert systems, plan recognition and Bayesian belief networks.

2.2.1 Uncertainty Management

The tutoring model in ITSs needs the student model to make pedagogical decisions and adapt instructions to the student’s needs. An increased level of adaptation requires building a more accurate student model, which is based on more information being available during interaction. Data gathered in the student model often includes uncertainty. One formal framework in artificial intelligence to deal with uncertainty is Bayesian Networks (BNs). Bayesian methods reason about the probability of future events, given their past and current probabilities. BNs enable computers to combine new data with prior beliefs about data, make subjective decisions about how strongly to weigh prior beliefs, and provide a policy for keeping new information in the proper perspective [29]. BNs are used in ITSs to support classification and prediction, model student knowledge, predict student behaviour, make tutoring decisions, and determine those steps when the student will need help and his/her probable method for solving problems [34]. Tutors decide among alternatives, within a probabilistic model of student knowledge and goals, which problem to present next.
2.2.2 Bayesian Networks

Bayesian Networks (BN) are graphical models designed to explicitly represent conditional independence among random variables of interest, and exploit this information to reduce the complexity of probabilistic inference [39]. A Bayesian Network is represented by a directed acyclic graph (DAG) and by a set of Conditional Probability Tables (CPT). Each node in this graph represents a random variable $X_i$ with associated domain $\text{dom}(X_i)$. Each arc represents direct dependency among these variables. If we associate with each node $X_i$ in the network a conditional probability table that specifies the probability distribution of the associated random variable given its immediate parent nodes $\text{parents}(X_i)$, then the Bayesian Network provides a compact representation of the Joint Probability Distribution (JPD) over all the variables in the network. Assuming that each node is conditionally independent of all its non-descendant nodes given its parents, then $p(X_1\ldots X_n) = \prod_{i=1}^n p(X_i|\text{Parents}(X_i))$ holds [37]. A joint probability distribution is a function $p$ on the Cartesian product of all the variables domain in BN such that the following two conditions hold:

1. $0 \leq p(v) \leq 1.0$, for each configuration $v \in V = \text{dom}(X_1) \times \text{dom}(X_2) \times \text{dom}(X_n)$,
2. $\sum_{v \in V} p(v) = 1.0$.

Conditional Independence is important in Bayesian Networks because it allows us to simplify large CPTs into small CPTs.

**Definition 2.2.1.1** Let $p(U)$ be a JPD, and $XYZ$ be pair wise disjoint subsets of $U$. $Y$ and $Z$ are conditionally independent given $X$, denoted by $I(Y,X,Z)$, if

$$p(Y|X,Z) = p(Y|X), \quad (2.1)$$
whenever \( p(X, Z) > 0 \). This conditional independence \( I(Y, X, Z) \) can be equivalently written as

\[
p(Y, X, Z) = \frac{p(Y, X).p(X, Z)}{p(X)}.
\] (2.2)

By combining conditional independence with chain rule, any JPD of a set of random variables can be defined in terms of a set of smaller CPTs. According to chain rule, a joint probability distribution \( p \) over \( U = \{A, B, C, D, E, F\} \) can be written as

\[
p(U) = p(A) \cdot p(B|A) \cdot p(C|A, B) \cdot p(D|A, B, C) \cdot p(E|A, B, C, D) \cdot p(F|A, B, C, D, E).
\]

Figure 2.2 shows a BN on variables \( U = \{A, B, C, D, E, F\} \).

![Figure 2.2: A BN on variables \( U = \{A, B, C, D, E, F\} \)](image)

Suppose the following conditional independencies among the variables \( A, B, C, D, E, F \) hold:

\[
\begin{align*}
p(C|A, B) &= p(C|A) & I(C, A, B), \\
p(D|A, B, C) &= p(D|B) & I(D, B, AC), \\
p(E|A, B, C, D) &= p(E|C) & I(E, C, ABD), \\
p(F|A, B, C, D, E) &= p(F|D, E) & I(F, DE, ABC).
\end{align*}
\]

By substituting into the chain rule, we obtain:
\[ p(A, B, C, D, E, F) = p(A) \cdot (B|A) \cdot (C|A) \cdot (D|B) \cdot (E|C) \cdot (F|D, E). \]

Specifying \( p(U) \) directly involves stating \( 2^6 - 1 = 63 \) probabilities. While using the BN, only \( 1 + 2 + 2 + 2 + 2 + 4 = 13 \) conditional probabilities need to be given. The BN form reduces the number of the model's parameters. Such a reduction provides great benefits for inference, learning (parameter estimation), and the computational perspective.

BN provide a more compact representation of the joint probability distribution over the variables of interest. Algorithms have been developed that compute the posterior probability of a variable given the available evidence on any other variable in the network. Although the complexity of probabilistic inference in BN is still exponential in the number of nodes, in practice it is often possible to obtain performance that is suitable for real-world applications. The underlying network structure facilitates the process of generating automatic explanations of the results of probabilistic inference, making BN very well suited for applications in which it is important that the user understands the rationale underlying the system behaviour, as is often the case for ITSs [57].

### 2.3 Eye Tracking

Eye tracking is the measurement of eye activities through an eye tracking device. We can measure either the position of the eye relative to the head or the orientation of the eye in 3D space. Almost all normal primate eye movements used to reposition the fovea (fovea is responsible for sharp central vision, which is necessary in humans for reading, driving, and any activity where visual detail is of primary importance) result as combinations of five basic types of movement: saccadic, smooth pursuit, vergence, vestibular, and physiological nystagmus (miniature movements associated
with fixations) [16]. Pursuit movements are involved when visually tracking a moving target. Depending on the range of target motion, the eyes are capable of matching the velocity of the moving target. Vergence movements are used to focus the pair of eyes over a distant target (depth perception). Optokinetic nystagmus is a smooth pursuit movement interspersed with saccades invoked to compensate for the retinal movement of the target. Vestibular nystagmus is a similar type of eye movement compensating for the movement of the head. Other movements such as adaptation and accommodation refer to non-positional aspects of eye movements (i.e., pupil dilation, lens focusing). The most common positional eye movements of interest in eye tracking are fixations and saccades. Fixations allow fovea to maintain a visual gaze on a stationary object of interest. Fixations are characterized by miniature eye movements: tremor, drift, and microsaccades. Our eyes are never still. In eye tracking, fixations represent the gaze position on the display and can be interpreted as areas of subject interest. Saccades are rapid eye movements used in repositioning the fovea to a new location in the visual environment. Saccadic movements are fast movements of both eyes in same direction. These movements occur between two fixations and can last from 10 ms up to 100 ms. Saccadic movements can be voluntarily executed or they can be invoked as a corrective measure (e.g., optokinetic or vestibular nystagmus ) [16].

There are four broad categories of eye movement measurement methodologies involving the use or measurement of: Electro-OculoGraphy (EOG), scleral contact lens/search coil, Photo-OculoGraphy (POG) or Video-OculoGraphy (VOG), and video-based combined pupil and corneal reflection [16].

The EOG technique relies on measurement of the skin’s electric potential differences of electrodes placed around the eyes. This method measures the position of the eye relative to the head. It is not suitable for measurement of the point of regard.
If we measure only the position of the eye relative to the head we cannot precisely determine if the subject is looking at the specific point unless we measure also the position of the head. The Scleral Contact Lens/Search Coil method involves attaching a mechanical or optical reference object mounted on a contact lens which is then worn directly on the eye. The principle method employs a wire coil, which is then measured moving through an electromagnetic field. This method also measures eye position relative to the head, and is not generally suitable for point of regard measurement. The POG and VOG category groups together a wide variety of eye movement recording techniques involving the measurement of distinguishable features of the eyes under rotation/translation, e.g., the apparent shape of the pupil, the position of the limbus (the irissclera boundary), and corneal reflections of a closely situated directed light source. In Video-Based Combined Pupil/Corneal Reflection, video-based trackers utilize relatively inexpensive cameras and image processing hardware to compute the point of regard in real-time. The apparatus may be table mounted or worn on the head. The corneal reflection of the light source (typically infra-red) is measured relative to the location of the pupil center.

2.4 Related Work

There are several existing projects using an eye tracker in a learning environment. Among the first projects to include eye tracking technology in e-learning is AdeLE (Adaptive E-Learning through Eye Tracking) [18]. The goal of this project is to develop an e-learning system which uses real-time eye tracking data to support adaptive teaching and learning. In the system described here [52], eye tracking data are used by an emphatic software agent to identify the learners’ focus and interest areas on the screen. The character agents are eye-aware because they use eye movements, pupil dilation, and changes in overall eye position to make inferences about the state of
the learner and to guide his behaviour. User studies showed that this type of agent could have beneficial effects on learner motivation and concentration during learning. e5Learning is another learning environment using eye tracking technology [9]. This system uses rectangular areas on screen to track users gaze position. Each of these Regions of Interest (ROI) has an associated time threshold predefined by the author of the course. The system then tracks each of these regions to determine whether or how those areas have been accessed by the user. The system developed in this project [14] detects student disengagement and boredom during the learning process. The system tracks student eye gaze patterns. When the student looks away from the screen for a certain period of time, the tutor assumed that the student was disengaged. These examples illustrate how eye tracking technology can be used to obtain a low-level mechanical account of cognitive processes during learning. However, none of those examples explored further pupillary response as a measure of mental workload. This issue presents an opportunity for research in which we can apply the eye tracking technology to further improve the student model. I would like to investigate whether the ITS can use mental workload measurements as additional information for assessment of the student knowledge state.

2.5 Summary

In this chapter I reviewed the ITSs technology, in particular I reviewed in more detail the Student Model including the uncertainty management. I discussed how Intelligent Tutoring Systems evolved from CAI and introduced problems which are the core of current research. Then I described each component of ITSs architecture: a knowledge domain model, tutoring model student model and user interface model. After this, I discussed eye tracking technology. This section discussed basic eye movement types which can be observed using an eye tracking device and also reviewed four types
of eye tracking techniques. At the end of this chapter, I reviewed current research in ITSs which are using eye tracking technology.
Chapter 3

MENTAL WORKLOAD AS MEASURE OF EFFORT

In this chapter, I discuss a method which presents a solution to the problem discussed in Chapter 1. I discuss pupillary response and its correlation to mental workload. By measuring mental workload we can determine if the student makes an effort thinking and therefore we can distinguish between different types of answers. This information can be used by the intelligent tutor to create a better representation of the student knowledge state. Later, I discuss how the Index of Cognitive Activity is used to measure the mental activity. At the end of this Chapter, I discuss the wavelet analysis which is used to analyse a pupil signal.

3.1 Mental Workload

Mental workload can be assessed with a number of techniques, including task performance on primary and secondary tasks [35], subjective ratings, and physiological measures (pupil size, heart rate, EEG) [20]. Pupil size is the most promising single measure of mental workload because it does not disrupt a user’s ongoing activities, provides real-time information about the user’s mental workload, and is less intrusive
than other physiological measures such as heart rate or EEG [24]. Pupil dilation is primarily the result of the integrated activity of two groups of muscles located in the iris. One set of muscles (the circular muscles) encircles the pupil; when activated, this set serves to constrict the diameter of the pupil and make it smaller. The second set of muscles (the radial muscles) lies immediately outside the circular muscles and extends radially from the pupil out through the iris. When activated, the radial muscles pull the pupil diameter outward and cause it to become larger. These two sets of muscles typically work together through reciprocal innervation, a physiological process involving both agonistic and antagonistic responses. Figure 3.1 shows both muscles when the eye is exposed to bright light (left) and dim light (right).

![Circular Muscles Contract
Radial Muscles Relax
Pupil Constricts](image)

![Circular Muscles Relax
Radial Muscles Contract
Pupil Dilates](image)

Figure 3.1: Radial and Circular Muscles.

Although the dilation of the pupil in response to increased attention was first observed early in the 20th century by Lowenstein, the first systematic study of the phenomenon appears to have been that of Hess and Polt in 1964 [21]. Under conditions of constant illumination and accommodation, pupil size has been observed to vary systematically in relation to a variety of physiological and psychological factors, including non visual stimulation, habituation, fatigue, sexual and political preference,
and level of mental effort [21]. The magnitude of the pupillary dilation appears to be a function of processing load, or the mental effort required to perform the cognitive task [24]. During measurement of pupil dilation it is important to exclude involvement of other potential confounding variables such light reflex or emotional processes which also cause variance in pupillary response. The chance that light reflex affects pupil dilation can be minimized by using displays with constant illumination. Emotional factors are well known for their expression in the autonomic nervous system. The effects of emotional arousal are generally longer lasting than the brief phasic responses evoked by cognitive activity [4]. In the work introduced here [4], Beatty showed that task-evoked pupillary response uniquely reflects the momentary level of processing load and is not an artefact of non-cognitive confounding factors. In fact, the task-evoked pupillary response has been widely used as a tool to investigate various aspects of human information processing, such as perception, memory, reasoning and reading [4]. Hess and Polt [19] demonstrated that there is, indeed, correlation between pupil dilation and problem difficulty. They showed that the pupil size gradually increased in diameter, reached its maximum immediately before the answer was given, and then returned to the previous control size. By comparing the mean size of the pupil of one subject, recorded five frames immediately before a question is asked, to the mean size of the pupil recorded five frames immediately before the answer is given, they were able to calculate the magnitude of increase in pupil diameter. In [24], Iqba, Zheng and Bailey used task-evoked pupillary response for the design of an attention manager that determines where in a user task sequence an application can gain user attention. The authors used percentage change in pupil size (PCPS), which is the measured pupil size at each task instant minus the baseline size, divided by the baseline. The average PCPS from the beginning to end of each task was used as the task-evoked pupillary response.

Using the knowledge about the correlation between the pupil dilation and the
mental workload, I developed a solution which gives the ITS the ability to consider the quality of the answers. By measuring the pupil dilation during different tasks, the intelligent tutor is able to recognise the amount of mental workload created by the student. The mental workload is seen by the intelligent tutor as the effort that is required to complete a given task, which in this case is the answer to a question. In the following section, I describe how we can measure the mental workload using the data created by an eye tracking device.

### 3.2 Measuring Mental Workload

Changes in light, no matter how miniscule, influence pupil size. During the presence of steady light, the pupil responds with continual but irregular oscillation. This movement is known as the light reflex. The reflex is fleeting, and the result is a visible pulsing of the pupil. When an individual experiences a psycho sensory stimulus (e.g. task requiring high cognitive workload), the pupil may make a response that is quite different from the light reflex. The radial muscles are activated and the circular muscles are inhibited. The result is a brief dilation that is greater than either muscle group alone could effect. For this reason, the phenomenon is called the dilation reflex [31]. The fundamental problem in studying the relationship between cognitive activity and pupillary response lies in how to separate the dilation reflex from the light reflex. Thus, the challenge has been to find a means of measuring the cognitive influence on pupil size while holding constant or removing the impact of light. More than a dozen years ago, researchers developed a technique for measuring cognitive workload based on changes in pupil diameter [31]. The Index of Cognitive Activity uses the signal processing techniques of wavelet analysis to detect small but reliable increases in pupil size while minimizing the impact of changes in light [31].
3.2.1 Index of Cognitive Activity

Eye movements and changes in pupil dilation provide important information about the user interaction with the display. Using eye tracking technology we are able to capture data in an almost continuous signal, providing precise information about user eye movements. Eye tracking devices are capable of capturing a large amount of data. For example using binocular settings with a sampling rate of 500 Hz, an eye tracking device is able to generate up to a million sample points during one session. This creates a challenge for the eye tracking researcher to reduce the data to a meaningful size. A new technique has been introduced that allows reliable and rapid estimation of cognitive workload from changes in pupil dilation. The technique is the Index of Cognitive Activity (ICA) [32]. The technique comprises of monitoring the pupillary response of the subject undergoing an evaluation of cognitive activity which includes at least one task, recording the pupillary response of the subject to the task, subjecting the recorded pupillary response to wavelet analysis in order to identify any dilatation reflex of the subject’s pupil during the task, and assigning pupillary response value to the result of the wavelet analysis [31].

To compute the ICA we need to collect data from the subject using an eye tracking device. Proper calibration and setup of the eye tracking device will ensure good quality of the collected data. Recorded data must undergo some pre-processing before we can apply wavelet analysis to pupil dilation data. The first step is to remove blinks from the recorded data. Each eye tracker has its own way to record blinks but usually blinks are represented by zero in the sample data. We need to remove all zero values from the data or use interpolation between two non-zero data samples before and after blink occurrence. Blinks (i.e., zero recordings) have been found to account for approximately 3-4% of all observation [31]. The second step is to remove partial blinks. Partial blinks must be eliminated because they create impossibly small pupil sizes in recorded data. Partial blinks account for another 1% of the total number of
observations. The result of removing blinks and partial blinks is a signal of precisely
the same length as the original pupil signal with the blinks and partial blinks replaced
with values that fall at equal intervals between the start and end points. When a
pupil signal is cleaned of blinks and partial blinks the next step is to perform wavelet
analysis to isolate dilation reflex.

3.2.2 Wavelet Analysis

Wavelet analysis is an exciting new method for solving difficult problems in mathe-
matics, physics, and engineering, with modern applications as diverse as wave propa-
gation, data compression, image processing, pattern recognition, computer graphics,
the detection of aircraft and submarines, and improvement in CAT scans and other
medical image technology. Wavelets allow complex information such as music, speech,
images, and patterns to be decomposed into elementary forms, called the fundamental
building blocks, at different positions and scales and subsequently reconstructed with
high precision [15]. Wavelet theory involves representing general functions in terms
of simpler, fixed building blocks at different scales and positions. Wavelet analysis
consists of repeated orthogonal transformation of the signal. The objective is to de-
compose the original signal into several independent components, each of which can
be analyzed and interpreted.

One criticism of wavelet analysis is the arbitrary choice of the mother wavelet
function, $\Psi_0(\eta)$, where $\eta$ is a nondimensional time parameter [49]. The mother wavelet
is a small function that is both oscillatory and that decays rapidly to zero in both
positive and negative directions (i.e., a little wave). For a given signal $x$ and mother
wavelet $\Psi$, the process of wavelet analysis is expressed by the formula:

$$\Psi_{jk}(x) = 2^{-j/2}\Psi(2^{-j}x - k),$$  \hspace{1cm} (3.1)
where \( j \) is an index of dilation and \( k \) is an index of translation and \( j, k \in \mathbb{Z} \). The systematic variation of indices \( j \) and \( k \) will create a family of wavelets able to fully reproduce the original signal [31]. Wavelet analysis proceeds iteratively: Using the mother wavelet function, the dilation transformation first extracts high frequency details from the signal by setting index \( j = 1 \) and evaluating all possible \( k \). Next, using a scaling function that is orthogonal to the wavelet function, a second transformation extracts from the signal all information not captured by the wavelet transform. The initial wavelet transforms the largest abrupt changes or discontinuities in the signal. The scaling transformation results in a smoothing of the signal because these discontinuities have been removed. The mother wavelet selected for isolating the dilation reflex from pupil signal is one from the Daubechies wavelets family [31].

**Example** Consider a signal recorded by an eye tracking device, which contains pupil dilation data recorded during the calibration process. Figure 3.2 shows the raw signal before pre-processing. We now remove blinks and partial blinks from the raw signal and perform wavelet analysis. Two levels of wavelet transformation are required before computing the ICA. After the first transformation we get a transformed signal which includes some noise. To remove the noise from the signal we set any wavelet value to zero which is above a certain threshold. After the second wavelet transform we get a smoothed signal which can be used to compute the ICA. Figure 3.3 shows the signal after the second wavelet transformation. Now we compute the ratio of unusual increases in pupil size observed during 10 samples by comparing the number of unusual increases against the theoretical maximum (which in this case is 10 as we check blocks of 10 consecutive samples in the signal). If we observed 5 increases in pupil size during 10 samples, the ratio would be 0.5. Each ratio is then scaled by a hyperbolic tangent function to a value which falls between 0 and 1. We compute the overall signal ratio by calculating the average of all ratios summed together and divided
by the number of ratios computed overall. As a result we get the Index of Cognitive Activity which then can be used by the system to evaluate the student’s effort. The above example shows the process of how to compute the ICA using a signal created by pupil dilations.

Figure 3.2: Raw Pupil Dilation Signal during calibration step.
Figure 3.3: Pupil Dilation Signal after two step wavelet transformation.

Figure 3.3 shows an signal after second wavelet transformation. This figure shows all frequencies which where used to produce the initial signal. If we combine these frequencies together we will be able to reproduce the original signal. Using the transformed signal we take block of 10 consecutive samples and watch for the change in frequency. Observing how many changes there are we can create and ration for each block of samples for entire signal. The overall ICA is then computed using this rations where we create an overall ration for all blocks created from this signal.

### 3.3 Summary

In this chapter, I presented a method which serves as solution to the problem where the ITS needs to recognise different types of an answer. I reviewed the effect of cognitive activity on pupil dilation. I discussed several research projects which demonstrated that there is in fact a correlation between mental workload and pupil dilation.
Pupil dilation is primarily the result of the integrated activity of two groups of muscles located in the iris. One set of muscles (the circular muscles) encircles the pupil; when activated, this set serves to constrict the diameter of the pupil and make it smaller. The second set of muscles (the radial muscles) lies immediately outside the circular muscles and extends radially from the pupil out through the iris. For the purpose of measuring the dilation reflex in a signal created by an eye tracking device, I described a technique based on wavelet analysis called the Index of Cognitive Activity. This technique is capable of isolating the dilation reflex from light reflex and assign value to the mental workload observed in the signal. At the end of the chapter, I described ICA computation in more detail.
Chapter 4

ARCHITECTURE OF PROPOSED SYSTEM

In this chapter, I introduce the Intelligent Tutoring Systems Measuring Student’s Effort During Assessment itself. I outline the major components of the system and describe how they interact with each other. I describe the student model which is represented as BN and how it is used in the system. I also describe the mental work-load assessment module and reading tracker. I have adapted the general architecture for ITS introduced in Chapter 2 to design this tutoring system. Each of the major modules is divided further into several subsystems, where each subsystem has its own functionality. Figure 4.1 shows the system architecture with each component from the general ITS framework divided into subsystems. All modules (Domain Knowledge, Student Module, User Interface Module), except the Tutoring Module, are implemented as separate modules. The Tutoring Module is implemented accross the entire system as opposed to being a single module. Details of each component are explained in more detail in the following sections.
In this subsection, I discuss the structure of the knowledge domain designed for this system. I also describe the assessment question set used by the system to obtain evidence about the student knowledge state.

4.1.1 Course Teaching Material

The knowledge domain in the system contains course notes, assessment questions and solution keys. Lecture notes are designed in such a way that the student can obtain information about concepts using a combination of text and pictures. The reason why I have included a graphical aid in the ITS is to accommodate each student’s preferred learning style [55]. I would like to support the learning preferences of a large group of students who might use this system. The domain focus is on teaching basic concepts.
about data structures, standard course material in the second year of a Computer Science program. To simplify the task of development of the ITS I have restricted the scope of the knowledge domain to six types of data structures: arrays, strings, lists, stacks, queues, and trees. Each type of data structure represents a separate topic. Each topic (e.g. arrays), is further divided in multiple concepts (subjects). To simplify the development even more I have restricted the division of topics to basic data structure manipulation concepts. Topics include concepts about adding, removing or accessing elements in data structures. Each concept is then composed of several pages which are used to deliver the required content to the student. Figure 4.2 shows an example of one data structure from the knowledge domain and how the topic is divided into concepts and how each concept is divided further into pages.

Overall, the knowledge domain in this system is represented using a tree like structure. In the Knowledge Domain Tree (KDT), each leaf node is linked to a page using an index key. Index keys are also used in the student model represented by BN to represent the student knowledge state. The KDT is entirely separated from the tutoring system. I have used XML to encode the course content into a separate xml file which is loaded by the program before each execution. This separation allows teachers to modify the course content without accessing and modifying the tutoring system itself. Figure 4.3 shows the entire KDT.
4.1.2 Assessment Question Set and Solution Keys

Assessment questions are used to assess the student’s knowledge state about each topic presented by the system. The Assessment Question Tree (AQT) is designed to mirror the KDT structure. One difference between the AQT and the KDT is that the AQT is extended by one more level of children nodes for each page node. Each leaf node in the AQT represents a question, and is marked using index keys. Questions use
multichoice answers and in some cases an answer is accompanied by a picture. The multichoice answer includes 4 choices from which 3 are incorrect and 1 is correct. The question stores, in a separate field, the correct answer. This field is used to compare the student’s response and determine if the response is correct or incorrect. Figure 4.4 and Figure 4.5 shows sample questions presented by the tutoring system to a student.

![Figure 4.4: Sample Question to Assess Student Knowledge.](image)

![Figure 4.5: Sample Question Including a Graphical Aid.](image)

As with the KDT, the AQT is also completely separated from the tutoring system.
As mentioned, I used XML to encode the AQT into a separate xml file which is loaded by the program before each execution. This separation allows teachers to create or modify questions without accessing and modifying the tutoring system itself. The AQT obtains evidence about the student’s concept mastery, which is then used to update the BN student model. In the next section, I discuss the student model which is used to represent the student’s knowledge state using BN. Figure 4.6 shows the entire AQT.
<table>
<thead>
<tr>
<th>TOPIC 1</th>
<th>SUBJECT ID</th>
<th>PAGE ID</th>
<th>QUESTION ID</th>
<th>TITLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>1.1</td>
<td>1.101</td>
<td>1.1011</td>
<td>Arrays are dynamic entities because they change size throughout program execution. Is this statement true or false?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.1012</td>
<td>Which from below options is a valid name for an array?</td>
</tr>
<tr>
<td>1.1</td>
<td>1.1</td>
<td>1.102</td>
<td>1.1021</td>
<td>How many elements are in array on this picture?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.1022</td>
<td>What is the value of 4th element?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.1023</td>
<td>What is the name of the array?</td>
</tr>
<tr>
<td>1.2</td>
<td>1.201</td>
<td>1.2011</td>
<td>1.2011</td>
<td>What is the correct declaration form for an array?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.2012</td>
<td>How can you additionally declare array size?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.2021</td>
<td>From the options below, which is the correct array initialization loop?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.2022</td>
<td>What values will be populated in numbers[] array for each elements after executing this loop?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.2023</td>
<td>From the options below, which is the correct initialization list of array?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.2024</td>
<td>If we omit the size of array from initializer list, what will happen?</td>
</tr>
<tr>
<td>1.3</td>
<td>1.301</td>
<td>1.3011</td>
<td>1.3011</td>
<td>What is the subscript/index used for in arrays?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.3012</td>
<td>What is the subscript/index of first element in an array?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.3013</td>
<td>What is the highest index in array declared as array[10]?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.3012</td>
<td>To access value of 7th element in array we use form</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.3012</td>
<td>How do you access any element in array?</td>
</tr>
<tr>
<td>1.4</td>
<td>1.401</td>
<td>1.4011</td>
<td>1.4011</td>
<td>From the options below, what is best example of 2D array?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.4012</td>
<td>How do you identify elements in table?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.4021</td>
<td>Which is the correct declaration form for 2 dimensional array?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.4022</td>
<td>What is the value of element b[2][3]?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.4031</td>
<td>How do you declare this array?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.4032</td>
<td>Which element has value 26?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.4041</td>
<td>How many subscripts you would use to access any element in 3D array?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.4042</td>
<td>What is the value of element a[0][1][2]?</td>
</tr>
<tr>
<td>2</td>
<td>2.1</td>
<td>2.101</td>
<td>2.1011</td>
<td>How would you declare an array to store string &quot;hello&quot;?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2.1012</td>
<td>What is the null character &quot;&quot; used for?</td>
</tr>
<tr>
<td>3</td>
<td>3.1</td>
<td>3.101</td>
<td>3.1011</td>
<td>What information are stored in each node of linked list?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.1012</td>
<td>Where is stored link to first node?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.1021</td>
<td>What is the main difference between array and linked list?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.1022</td>
<td>How do you find specific node in linked list</td>
</tr>
<tr>
<td>3.2</td>
<td>3.201</td>
<td>3.2011</td>
<td>3.2011</td>
<td>How is the first node called?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.2012</td>
<td>How is the last node called?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.2021</td>
<td>When the traversing of list stops?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.2022</td>
<td>Which is easier for inserting into linked list?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.2023</td>
<td>After inserting node 5 (shown in picture) what will the link in head point to?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.2031</td>
<td>What will node 3 link point to after removing node 5 from list?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3.2032</td>
<td>What will node 5 link point to after removing node 2 from list?</td>
</tr>
</tbody>
</table>

Figure 4.6: Assessment Question Tree.
4.2 Student Module

The student model discussed in this thesis is composed of two parts. The first part consists of statistical measurements which include the student’s performance scores, concept completeness, time spent on questions, and reading scores. The second part is a probabilistic student model based on a BN. BNs have been applied in student modeling several times before [22] [55]. One of the distinguishing features of an ITS is that it is capable of adapting its instruction to the specific needs of each individual student, as good human tutors do [37]. The more an ITS needs to know about the student to provide the desired level of adaptation, the more challenging it is for the ITS to build an accurate student model.

4.2.1 Building student model using a Bayesian Network

To build the directed acyclic graph (DAG) of a BN, I had to first identify all random variables with dependencies, which are represented in the graph as nodes (random variables) with directed edges (dependencies) between nodes. For my purpose, I have identified a set of concepts from the domain of data structures which I am using in this tutoring system to teach students. Each concept is then represented in the DAG as a node. I have also identified dependencies between these concepts, which represent the prerequisite relationship. For example, the concept of adding a new element to an array cannot be learned before the student first learns how an array is structured. These dependencies can be represented clearly and directly in the DAG by adding directed edges from one concept to another where knowledge of the former is a prerequisite for understanding the latter. Each node in the DAG is a random variable which can take only two possible values 0 or 1, or in this case, each node can take two possible values known and not known. We can estimate the student’s knowledge on one concept by calculating the $p(\text{concept} = \text{known}|\text{evidence})$, where
the evidence is the already observed information about this student using Students Knowledge Assessment. To model the student’s knowledge for the whole domain, we use a joint probability distribution:

\[ p(\text{concept}_1|\text{evidence}, \text{concept}_2|\text{evidence}, ..., \text{concept}_n|\text{evidence}) \]

where \( \text{concept}_1,...,\text{concept}_n \) are the concepts of the knowledge domain and \( \text{evidence} \) is what system observes from the student. Figure 4.7 shows a small portion of the BN designed for this system.

![Bayesian network model for queue data structure concepts.](image)

After building the DAG, we need to specify CPTs for each node given its parents. Due to various difficulties, privacy in particular, I could not use any previous results from exams to construct my CPTs. To solve the CPTs acquisition problem I have used the Maximum Likelihood Estimation (MLE) method. MLE allows me to compute the probability for each node by using observed evidence. Each node (concept) in the BN has its probability distribution as:

\[
\text{node(known)} = p \\
\text{node(not known)} = 1 - p
\]  

(4.1)

To compute the likelihood \( L \), I use the form:

\[
L = p^K (1 - p)^U,
\]  

(4.2)
where $K$ is the number of trials in which the concept is known and $U$ is the number of trials in which the concept is not known. In order to get the MLE we need to compute the log-likelihood which is done using this form:

$$
\ln L = K \ln(p) + F \ln(1 - p),
$$

(4.3)

We then take the derivative of this with respect to $p$

$$
\frac{\partial \ln L}{\partial p} = \frac{S}{p} - \frac{F}{1 - p}.
$$

(4.4)

We now set this equal to zero and solve for $p$:

$$
\frac{S}{p} - \frac{F}{1 - p} = 0
$$

$$
S(1 - p) - Fp = 0
$$

$$
p = \frac{S}{S + F}
$$

Using the MLE, I can determine the probability of each concept being known, namely $p(\text{concept}_i = \text{known})$. Moreover, I can compute the probability of concept and the prerequisite concept using $p(\text{concept}_i = \text{known}, P_{\text{concept}_i} = \text{known})$.

From $p(\text{concept}_i = \text{known}, P_{\text{concept}_i} = \text{known})$, the desired CPD $p(\text{concept}_i = \text{known}|P_{\text{concept}_i} = \text{known})$ can be obtained through the following equation:

$$
p(\text{concept}_i = \text{known}|P_{\text{concept}_i} = \text{known}) = \frac{p(\text{concept}_i = \text{known}, P_{\text{concept}_i} = \text{known})}{p(P_{\text{concept}_i} = \text{known})}.
$$

(4.5)

Utilizing the above, I can construct every CPT for the entire BN. Figure 4.8 shows the entire BN. Figure 4.9 shows the same BN implemented as a 2D array.
Figure 4.8: Bayesian Network Student Model.
Figure 4.9: Bayesian Network implemented as 2D array.
4.2.2 Updating student knowledge state

For the tutoring system to be able to adapt teaching strategies to student needs, it is necessary for the tutoring system to keep the student model updated based on student responses. The student model is updated after each assessment of the student knowledge state. As previously discussed, each concept from the knowledge domain is represented in BN by a node, which is a binary variable taking only two possible values; the concept is either known or not known. To update the node as known, the student must reach a satisfactory score in his/her knowledge assessment. The score must be greater then a predefined threshold to meet the requirements of the required concept. I discuss in more detail the calculation of the score in Section 4.5.2. After observing the evidence and updating the concept knowledge state, the entire BN is updated. After the entire network is updated, the tutoring system can use this network as input to make the best suitable teaching decision.

So far, I have discussed the knowledge domain used by the tutoring system to introduce concepts from the domain of data structures. I explained how this knowledge is represented and also how it is assessed to obtain evidence about the student’s knowledge state. This evidence is used in the student model to update the student probabilistic model. The updated student model is used by the tutoring system to adapt the learning environment to the student’s current needs.

4.3 Eye Tracking Device

The eye tracking device used in this project is the SR Research model EyeLink II. The EyeLink II has the highest resolution (noise limited at < 0.01°) and fastest data rate (500 samples per second) of any video-based eye tracker today. On-line gaze position data is available with delays as low as 3 milliseconds, making the system ideal for gaze-contingent display applications. In addition, on-line data parsing occurs, making
eye events such as saccade, fixation, and blink available within 25\(ms\) to the display computer [48]. The EyeLink II system consists of three miniature cameras mounted on a leather-padded headband. Two eye cameras allow binocular eye tracking or easy selection of the subject’s dominant eye without the mechanical reconfiguration required by most head mounted eye trackers. Figure 4.10 shows a typical EyeLink II system configuration.

![Diagram of EyeLink II System Configuration](image)

Figure 4.10: Typical EyeLink II System Configuration [48].

Before each trial, the student needs to setup the eye tracking device in order to get good quality data. The setup consist of placing the headband on the student’s head so that the headband does not move in any direction. Both cameras need to adjusted so that the pupil is centered in the camera window. Figure 4.11 show correct adjustment of the camera tracking one eye.
After this, the system runs a set of calibration and validation procedures to ensure that the recorded data is correct. In the calibration step, the student is required to pursue with his/her eyes a target displayed on the screen. The target moves in random directions. During this movement the EyeLink II records data from the student’s eye positions. After the calibration step the system validates recorded data by running the same test again. If the EyeLink II system setup is correct, the tutoring system is loaded into the display PC. During EyeLink II operation, the tutoring system is capturing online data for each content presentation session and for each assessment question session. Each recording captures gaze position data and pupil size data. In Appendix A is included the method which is implemented in this tutoring system to access the latest data produced by the eye tracking device. I decided to use $250\,Hz$ sampling rate for this project. In Section 4.5.2, I discuss in more detail how the data generated by the eye tracker are used to assess the student’s mental workload.

4.4 User Interface Module

The interaction between the student and the tutoring system is managed by the User Interface Module (UIM). The UIM displays tutoring instructions to the student and
captures the student’s input. I have developed the UIM using the Simple Direct-
Media Layer [46]. The user interface is operating in several states, which depend on
the current state of the tutoring system. After the tutoring system has loaded all
necessary data, the UIM presents the main screen with the current curriculum. In
this screen, the student can choose which topic he/she wishes to know more about.
Figure 4.12 shows the main screen after the tutoring system has been launched. By
clicking on the desired topic, the tutoring system presents the desired content. The
UIM changes with the tutoring system state and presents new content to the student.
Figure 4.13 shows the screen after selecting the concept Multi dimensional arrays
from topic Array. The entire UIM is controlled by mouse events triggered by the
student. I have tried to avoid any input which requires the keyboard to eliminate
as much unnecessary head movement as possible. By using the keyboard input, the
student is forced to move his/her head downwards towards the keyboard to select the
right input key. The goal was to ensure that the student’s eyes are gazing upon the
screen for the duration of the session so the data produced by the eye tracking device
are of the best quality possible.
### 1.4 Multi dimensional arrays

Below are concepts you need to be familiar with in order to complete this section.

<table>
<thead>
<tr>
<th>Concepts</th>
<th>1.401</th>
<th>1.402</th>
<th>1.403</th>
<th>1.404</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instructions</td>
<td>Basic introduction of concept</td>
<td>Accessing array</td>
<td>Example 2D array</td>
<td>Example 3D array</td>
</tr>
<tr>
<td>Complete</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Before you can start this section you need to finish below prerequisites first.

- 1.101 Basics Definitions
- 1.102 First Example of an int array
- 1.201 Declaration of array
- 1.202 Initialization using loop
- 1.203 Initialization using Initializer List
- 1.301 Array element subscript
- 1.302 Accessing array

Sections highlighted red are not completed yet.

---

**Figure 4.13: Concept Screen.**
4.5 Tutoring Module

The tutoring module is the heart of this system. This module adapts the learning environment to the student’s current needs based on input data collected during his/her interaction with the system. To describe the function of the tutoring module, I explain its functionality in two sections. Section 4.5.1 discusses the system’s capability to adapt, and Section 4.5.2 discusses the tutoring module data acquisition.

4.6 Adaptive Learning Environment

The adaptation model discussed in this tutoring system was inspired by the work of Hua [22]. Using a student model represented by BN, three different types of system adaptation emerge: Best Topic To Learn, Concept Prerequisites, and Learning Path Generator. In following sections, I describe each of the above techniques in more detail.

4.6.1 Best Topic To Learn

This type of adaptation is based on the idea of already having most of the required knowledge in order to understand a certain concept. Using a probabilistic student model, the tutoring module can find a concept in the knowledge domain which will be the easiest to understand, based on knowledge already learned by the student. When this concept is found, the system highlights it for the student, so he/she has the option to follow the tutoring system recommendation. I left the option to follow the Best Topic To Learn recommendation to the student, as I wanted the student to remain in complete control over the learning process. Each time the student model is updated with new information, a new Best Topic To Learn is found and recommended to the student. Figure 4.14 shows part of the tutoring system screen with Multi dimensional arrays highlighted as the Best Topic To Learn.
4.6.2 Concept Prerequisites

To explain this technique of adaptation, I start with an example. A student wants to learn about the concept of removing an element from a stack data structure. Before the student can learn the stack method pop, he/she needs to understand how a stack data structure is defined. The concept of stack definition is pre-requisite knowledge to understand the stack method pop. Relationships between these two concepts are captured as dependencies in the BN student model. Using this property we can build a list of pre-requisites for each concept in the knowledge domain. Following parent nodes of each child node, using edges from each child to its parent we can obtain a list of dependencies among all nodes in the BN. The tutoring module builds a pre-requisite list for each concept selected by the student and displays it using the User Interface Module. The pre-requisite list for each concept must be first learned.
by the student in order to continue with the next concept. This restriction will guide the student through the right sequence of learning steps. Figure 4.15 shows the concept of Removing an element from Queue with all pre-requisites. Each pre-requisite highlighted in red must be learned first before the student can continue with the selected concept.

5.3 Removing element from Queue

Below are concepts you need to be familiar with in order to complete this section.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Score</th>
<th>Complete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Locked</td>
<td>0.0</td>
<td>0.0%</td>
</tr>
</tbody>
</table>

Before you can start this section you need to finish below prerequisites first.

1.101 Basics Definitions
1.102 First Example of an int array
1.201 Declaration of array
1.202 Initialization using loop
1.203 Initialization using Initializer List
1.301 Array element subscript/index
1.302 Accessing array element
4.101 Basics Definitions
4.201 Push - Stack using array
4.301 Pop - Stack using array
5.101 Basics Definitions
5.201 Inserting into queue

Sections highlighted red are not completed yet.

Figure 4.15: Removing an element from Queue with all pre-requisites.

4.6.3 Learning Path Generator

Building a learning path utilizes the technique of Concept Pre-requisites. The student has the option to select each topic as a learning goal. For example, a student may wish to learn everything about the tree data structure before the assignment deadline.
With limited time, the student wants to learn the selected topic in the minimum number of steps possible. The tutoring system generates a pre-requisite list for each concept in the selected goal and combines these lists into a learning path sequence. Concepts already learned by the student are omitted from the learning path list. This technique of adaptation ensures that student can learn in the most efficient way.

4.7 More Details For Better Effectiveness

Human teachers learn about students through years of experience. Master teachers often use secondary learning features, (e.g., a student’s facial expression, body language, and tone of voice) to augment their understanding of a student’s learning. Interaction between students and teachers provides critical data about the student’s goals, skills, motivation, and interest. Similarly, intelligent tutors make inferences about presumed student knowledge and store this information in the student model [37]. For the teacher to adapt teaching instruction to the student’s needs, the teacher needs to first observe the student’s behaviour. The more the teacher can observe, the better he/she can understand what a student may need in a given moment. More details about the student can lead to more effective adaptation of teaching instruction. In this section, I discuss which data are observed from the student’s interactions with the system, and how these data are used by the tutoring module to create a more effective adaptive environment.

4.7.1 Reading Tracker

During the estimation of the student’s knowledge state using the question-answer technique, one important question is to determine if student has the necessary knowledge to answer the question. Paying attention is the first step in the learning process. For example:
While the teacher explains new topic to the students, some of them are not paying attention to what is described in the class. As a result of not listening to the directions in class students may make errors on their assignments because they missed part of the presented knowledge.

It is important for the tutoring system to know if the student is paying attention to what is being currently presented. Several techniques have been used to infer if the student accessed the necessary learning material. Zakka [28] used a mouse to select a rectangular area on the screen to define a region of presented knowledge. This process requires the teacher to define each region manually before the tutoring system starts. The disadvantage of this technique is that each region must be selected manually by the teacher. On the other hand the teacher can define the order of the regions which must be accessed by the student. Calvi, Porta and Sacchi [9] used the concept of Region Of Interest to define each part of the concept on the screen. This technique also uses rectangular areas predefined on the screen. These areas are then used to track student gaze patterns. Both techniques are unable to tell the tutoring system if the student in fact read the presented concept, because both techniques track only if the gaze position of the student falls within the boundaries of predefined rectangular regions. The student could look at one word within this predefined region for two minutes and both techniques would indicate that the student has accessed the necessary material to answer the follow-up question. The technique developed in this thesis uses a more precise method of measuring student reading activity.

The idea behind this technique comes from collision detection in games. To detect if two spheres collide in 3D space, we can use the Euclidean distance formula:

\[
distance(s1, s2) = \sqrt{(x2 - x1)^2 + (y2 - y1)^2 + (z2 - z1)^2},
\]  

\[ (4.6) \]
where \( x, y, z \) are coordinates in 3D space. We then compare the distance to the sum of both spheres’ radii. If the distance is less than, or equal to the sum of the radii, then the two spheres collide. ITS content is presented on a flat screen which is a 2D plane so we can omit the \( z \) coordinate from the formula (4.6) and compute the distance using only \( x \) and \( y \) coordinates between two circles. These two circles are created using eye gaze coordinates and word position on screen. So how do we position each word on the screen? Each concept presented to the student is loaded as a continuous

```
Input: Current Page Index Key CPIK
Output: List of words including their position coordinates
Initialize temporary variables to store values for row height, gaps and screen size
Clear list with text
for all topics in sylabus do
    for all subjects in each topic do
        for all pages in each subject do
            get page index key PIK
            if PIK == CPIK then
                Clear list with words
                Update list with words by loading text from the current page
                for all words in list with words do
                    compute word length
                    set word x coordinate = gap between words + word length * font size
                    set word y coordinate = row
                    add word to list with text
                end for
                row+=font size + gap between words
            end if
        end for
    end for
end for
```

Algorithm 1: Algorithm which generates position coordinates for each word.

stream of words from the Knowledge Domain Tree (see Section 4.1). This stream is used to compute the exact position of each word on the screen. When placing words on the screen, the system also stores the word’s position in a separate data structure. Algorithm 1 shows the steps which are required to compute the screen coordinates for each word. By retrieving the correct page content, the program is able to access text
contained in this page. The text is stored as a continuous list of words in a list data structure. Iterating through this list, we are able to access each word and compute the length of each word. The length is used to compute the $x$ coordinate of the word where the height (font size) is used to compute the $y$ coordinate.

During a reading session, the student reads content displayed on the screen and the eye tracking device sends data about the student’s gaze position to the tutoring system. The tutoring system then computes the Euclidean distance for each word which has not been read by the student. Each word which is in collision with the new gaze position data is flagged as a word read by student. Appendix A contains part of the code which is used by the tutoring system to mark each word as read by the student. At the end of the reading session the tutoring system counts how many words have been read and calculates the percentage of covered content. If the percentage of covered content is less than a predefined threshold (in this system it is set to 70% ), the system will understand this result as a failure of the student to read the required content. The result from the Reading Tracker is then used in Student Effort Assessment to produce a more precise representation of student knowledge for the given concept.

4.7.2 Mental Workload Assessment

The uncertainty in student responses represents a problem for any ITS, especially the uncertainty created while answering questions. To describe the problem in more detail I provide the following example:

During the midterm exam, the students are required to use an answer sheet which is composed of multi-choice questions. The student must select one answer from the possible choices to each question. John, as a responsible student, has prepared himself for this exam by studying the whole week. John was able to select the correct answer for each question in the answer sheet using the knowledge he gained and his
final score was 100%. Steve, on the other hand, was a less responsible student and he spend the last week in drinking and visiting parties. Steve’s knowledge about the concepts is very limited, therefore he took a guess on every question. But Steve’s luck helped him to score 100%. The teacher has real difficulty determining who from all the students understands the assessed concept because the only indication about the student’s knowledge state is the score.

It is worth mentioning that even if the teacher had access to results from previous exams to determine the validity of the current scores, the teacher would still have a problem telling if the answers are just guesses, or were produced using gained knowledge. Therefore, it is important to take into consideration the quality of an answer. Answering a question requires a student to apply learned knowledge in practice or recall facts from memory. These processes have an effect on cognitive workload, which can be measured using the ICA. The ICA can be used to estimate how much effort was taken by the student while answering a question. As described in Chapter 3, Hess and Polt [19] demonstrated that there is a correlation between pupil dilation and problem difficulty. When the task gets more difficult, the cognitive workload increases which is then reflected as increased pupil dilation. They showed that the pupil size gradually increased in diameter, reached its maximum immediately before the answer was given, and then returned to the previous control size (see Chapter 3 for more details).

The tutoring system, discussed in this thesis, uses ICA to measure student effort during assessment and uses the result to update its belief about the student’s knowledge state. How does the system knows that the student made an effort in answering a question? The process of determining this involves two steps of ICA computation. The first ICA is computed during the calibration step before the question is presented. The student is asked to remain still and look on a blank screen for a period of 10 seconds. During this time, the tutoring system records pupil dilation
and produces the first ICA called Calibration Pupil Size (CPS). The result from this step is used as the base for later comparison. I assume that this CPS represents low mental workload because the student is not solving any problems which require cognitive activity. Algorythm 2 shows how to compute the ICA from the raw signal. After the calibration step, the tutoring system presents a question to the student and starts a new recording of pupil size. When the student selects an answer to the presented question, the tutoring system produces a second ICA called the Workload Pupil Size (WPS). The tutoring system compares the WPS to the CPS to determine if there is an observable increase (i.e., $WPS > CPS$) in mental workload. The result of the mental workload assessment is used by the Student Effort Assessment module as another input to produce a more precise representation of student knowledge for the given concept.

```
Input: Raw signal
Output: ICA
Initialize temporary variables
for all samples in raw signal do
    if pupil size < 0 then
        interpolate pupil size values to remove blinks and partial blinks
    end if
end for
transform signal using wavelet tranform
for all samples in processed signal do
    update each sample which does not fall within the range of possible values to 0
end for
transform signal using wavelet tranform
for all samples in processed signal do
    compute the ratio of maximum increase per block of samples
end for
ICA=compute the ratio of maximum increase per signal
```

Algorithm 2: Algorithm for computing ICA.
4.7.3 Student Effort Assessment

The student knowledge state for a given concept is estimated by the *Student Effort Assessment* module (SEA). The SEA uses the AQT, discussed in Section 4.1, as input to generate questions and obtain answers from a student. The second input to SEA is created by *Reading Tracker*, which determines whether the student covered the required concept for the given question or not. If student did not read the required concept, he/she may not answer the question correctly. The third input used by SEA is provided by the *Mental Workload Assessment* module, which is based on the ICA discussed in Chapter 3.2. These three inputs are seen by the system as the effort which the student gives in answering a question about a specific concept. Using these three inputs, I can compute the probability of the student’s understanding of the current concept and the update student’s knowledge state. Figure 4.16 shows the BN build for assessing the student knowledge state using these three inputs from the student.

![Bayesian Network for assessing student knowledge state.](image)

CPTs are constructed using the MLE method (see Section 4.2). All variables in this BN can take only two possible values; *true* or *false*. The result from the SEA is the probability representing the degree to which the student gained new knowledge of a given concept. If the probability is higher than a predefined threshold (e.g., \( p(\text{new knowledge}) > 0.7 \)), the tutoring system updates the student model with this
new evidence. Unlike most conventional assessment (testing), which measures how much a student knows, the Student Effort Assessment measures effort to find out if the student learned. Using this new method of estimating the student knowledge state, the tutoring system is able to create a more detailed student model, and consequently use the student model to adapt the learning environment in a more effective manner.

4.8 Summary

In this chapter, I have introduced the general architecture of the ITS measuring student effort using eye tracking technology. I outlined the major components of the system and described how they interact with each other. The chapter continued by describing the student model which is represented as BN and how it is used in the system. After this, I introduced the eye tracking device used in this research. At the end of this chapter, I discussed the tutoring module and how a student’s knowledge state is assessed using multiple inputs from student interaction. A more detailed assessment of the student’s knowledge state creates the opportunity for the tutoring system to adapt its learning environment in a more effective way.
Chapter 5

DISCUSSION

In this chapter, I describe in detail how students may use the system for learning. I also describe the operation of the intelligent tutor during the learning session. At the end of this chapter, I describe a planned user study.

5.1 System Usage

The tutoring system discussed here is very easy to use. Anyone who has basic computer skills can employ this system for learning. The system starts with a setup of the eye tracking device, which is required for measuring the student effort. The tutoring system can work without the eye tracking device but the functionality will be limited to operation similar to an earlier system developed here [22]. During the setup process, each student is required to follow the instruction presented by the system. The whole process takes about five minutes to complete. After the setup process, the tutoring system presents the syllabus to the student. At this point, students have an option of following the recommendation made by the system, or they can choose what topic is the best to learn for them. As discussed in Section 4.4, the student can use a mouse to navigate and select any topic presented on the progress screen. Each topic is displayed in the form of rectangular icon, which includes the topic name,
score, percentage of completed concept and status which is accompanied with the aid of color.

The status gives the student information about the topic accessibility. If the topic was already learned by the student, the status will be set as “Done” with green color as the background. If all the prerequisites have been met for a given topic, but the topic is still not known by the student, the status will indicate that this topic is in “Progress” with orange color as background. This topic can be accessed by the student for learning. If there is at least one prerequisite for a topic which has not been completed, the given topic is not accessible to the student and the status will show that this topic is “Locked”, accompanied with red color background. Following these status icons, the student will have a fair understanding about what topics he/she has already accessed, and can choose which topic he/she wants to learn next.

When the student selects a new topic, the system presents the student with more information about the topic by loading the subject screen. When the subject screen is shown, the student can see what the prerequisites are for the selected topic, and what pages are available to learn. At this point, the student has an option to select this topic as the learning goal. When the student decides and selects this topic, the tutoring system will generate the learning path and will present the learning content to the student in a sequence.

The other option for the student is to access each individual page of this topic, but only if all required prerequisites have been completed. By accessing only an individual page, the tutoring system will present only the content for the selected page. If the student selects a page with the status “Locked”, the tutoring system will remind the student that there are still some prerequisites which have not been completed yet. The tutoring system then asks the student to first complete the required prerequisites before the student can proceed to this new concept. After all requirements have been met for a selected page, the tutoring system will load a page screen.
During the loading process, the tutoring system will perform the drift correction operation to make sure that the data which are recorded by the eye tracking device are accurate. The drift correction step requires the student to gaze at one mark in the middle of the screen for period of 10 seconds. After this step the tutoring system will present the page screen.

This page screen now includes the concept which the student needs to learn. The concept is sometimes accompanied with a figure, which helps the student to understand the concept better. The student reads the presented content and the tutoring system records the reading pattern of the student. When the student is done reading the content, he/she has the option to exit this page or let the tutoring system assess the student. When the student decides to be assessed, the tutoring system will exit this page screen and perform another drift correction step. During this drift correction step, the tutoring system also records the pupil size which is later used as the base for computing the ICA. After recalibrating, the tutoring system presents a question to the student, and starts another recording of eye activities. The student now can read the question and select one of the four presented answers. The student makes the selection using the mouse. When the student selects an answer, the tutoring system evaluates the response and provides feedback to the student. At this point, the tutoring system has already computed the ICA, evaluated the reading pattern and updated the Bayesian Network student model with this new information. The student can now exit the assessment screen if there are no other questions or select the next question to answer.

After the assessment step the student is directed back to the page screen where he/she can make a selection of another page, or return to the progress screen and select another topic. When the student is following the learning goal path, the tutoring system is generating all pages with assessment questions in sequence so the student does not need to return to the page screen and select the content manually. The
student can repeat the same process for each topic and pages until the student achieves mastery in the entire syllabus. The tutoring system provides an option to repeat each topic so the student can access the content over again. After completing selected topics, the student can exit the system and save the progress for later analysis. All data recorded by the tutoring system are stored in a local file system. The data produced by the eye tracking device are stored in an EDF file in the eye tracking device computer.

5.2 System Advantages

The tutoring system discussed in this thesis is a real working tutoring system. Currently, this system is deployed and operates in the Regina Integrative Cognitive Experimentation Lab, where the eye tracking device is located. By utilising the data from an eye tracking device this system is capable of capturing the student effort during the learning and the assessment. This functionality makes this tutoring system unique, because it allows the tutoring system to accomplish tasks which are not possible by a human teacher.

5.2.1 Tracking the reading pattern of the student

The new method of tracking the student’s reading pattern is capable of recognising to what degree the student had read the presented content. This ability gives the tutoring system more precise information about the student’s behaviour. There already has been some effort in this area [9, 28], but this new method is superior to previous methods because it is more precise, dynamic and does not require any input from the teacher to define the regions of interest. The advantage of this method is that it is treating each word as an individual piece of knowledge as opposed to previous methods which have considered the entire block of text as one piece of knowledge.
The method of the tracking student’s reading pattern can be also used to track a graphical aid content to determine if the student follows the pictures on the screen. Similarly to text content, the tutoring system will treat a picture as a single piece of knowledge and then track the gaze position of the student. This detailed segmentation of the content allows the system to build a more precise model of the student’s reading pattern. This model is used by the tutoring system to conclude how much of the student’s effort was given to read a presented concept and as a result of this the tutoring system can better estimate the student’s knowledge state.

5.2.2 Distinguishing between a guess and valid answer

Another advantage of this system is its ability to measure mental activity of the student during an assessment. To measure the mental activity, the tutoring system uses an eye tracking device to record changes in the pupil size during an assessment. Researchers have known about the correlation between mental activity and pupil dilation for decades, but this knowledge has never been used in Intelligent Tutoring Systems for the purpose of measuring the student effort. The method introduced in this thesis is novel and unique to this tutoring system. When the student formulates an answer to a question, the student uses the knowledge previously learned. Accessing this knowledge in memory requires an increased mental activity which can be detected using pupil dilation. This increase in mental activity is seen by the tutoring system as an effort. By measuring the effort which is used by the student when answering a question, the tutoring system can distinguish between an answer which is produced by careful thinking and an answer which is just a guess. The advantage of this method is that the tutoring system is no longer considering only if the answer is correct, but also if the correct answer is a guess or not. When an answer is correct but it is recognised as a guess, the tutoring system can assume that the student may still have some problems understanding a new concept. Based on this assumption the tutoring
system can make a decision to re-assess the student or to suggest the student go back and read the assessed concept again.

As discussed in Chapter 4, the system is able to assign different probabilities to the student’s response because the system can now distinguish between a guess and a valid answer. Combining this method with the method of tracking a student’s reading pattern, the tutoring system is capable of creating more precise representation of the student’s knowledge state which leads to more effective adaptation of the learning environment.

5.2.3 Dealing with uncertainty in the student’s responses

The tutoring system discussed in this thesis is capable of making decisions under uncertainty. Using the information explicitly available during interaction the tutoring system is able to create a partial representation of the student. The student model in this tutoring system is represented using a BN. BN enables the ITSs to combine new data with prior beliefs about data, make subjective decisions about how strongly to weight prior beliefs, and provide a policy for keeping new information in proper perspective. In this tutoring system, the BN is built from nodes which are all observable, therefore the entries for the network’s CPT can be learned using the MLE from frequency data. The advantage of a formal probabilistic approach is that the model only needs to quantify local dependencies among variables. Another advantage is that the BNs support decision making approaches that rely on the sound foundation of decision theory. The BN built to represent the student’s knowledge state is used by the tutoring system to make decisions which are seen as the best solutions at a given time and help the tutoring system to adapt instructions to suit specific needs of the student. The ability to adapt teaching instruction to student needs is one of the distinguishing features of an ITS.
5.2.4 Tutoring system independent of the knowledge domain

The knowledge domain is made of teaching material and an assessment questionnaire. Both parts are used by the tutoring system to teach the students new concepts from a knowledge domain. Following the typical modular architecture of an ITS, the knowledge domain is built as a separate part of the tutoring system. By separating the knowledge domain from the tutoring system, it becomes easy to update the tutoring system with new content. This feature is beneficial to the teachers because a teacher does not need to have any knowledge about the tutoring system’s internal mechanics in order be able to update the knowledge domain. This separation allows multiple teachers to work simultaneously on one knowledge domain. Another advantage of this approach is that the separation allows the concepts to be indexed and retrieved efficiently, allowing the system to adapt to one particular student’s needs and knowledge state.

5.2.5 Dynamic user interface

The user interface in this tutoring system is designed in such a way that it can accommodate any amount of teaching material contained in the knowledge domain of the tutoring system. Using the angular distances, the user interface can resize itself to the size of a screen on which it is displayed. This advantage makes the user interface fit any screen size or resolution. Although total screen size independence is not possible because of the font size, the dynamic resizing will ensure that the user interface looks the same way at all times. Lastly, the user interface design is simple, which allows easy navigation and immediate access to information of interest. A well designed interface can enhance the capabilities of an ITS by allowing the system to present instructions and feedback to the student in a clear and direct way.
5.3 Planned user study of the tutoring system

As part of my research, I intended to perform a user study of this tutoring system. I planned to have two different groups of users who would use this tutoring system. The first group would use the tutoring system which also uses the eye tracking device to measure the student’s effort. The second group of users, the control group, would use the same tutoring system with the eye tracking device but the tutoring system would not use the eye tracking device data for measuring the student’s effort. The second group would not know that the eye tracking device is not operational during the learning session. After all groups had finished the experiment, I planned to analyse the data with ANOVA to see if there is any significant difference between these two groups of users. I received approval for this research from the Ethics Research Board (See attached form in Appendix B). Unfortunately, I was unable to recruit any participants for this research and perform any analysis to see how effective this new tutoring system is.
Chapter 6

CONCLUDING REMARKS

6.1 Conclusions

In this thesis, I proposed a new Intelligent Tutoring System for an introductory course in data structures. I discussed a new architecture for an Intelligent Tutoring System which utilises data produced by student eye activities. Explaining the structure and content of each component of the architecture, I described the concept and the uniqueness of this tutoring system. A fundamental problem in assessing student knowledge is the problem of assigning correct credit to the each answer. Using the question-answer technique, an ITS can obtain information about the student knowledge state. When using this technique of assessment we introduce uncertainty because the response from the student to a question is not always the same. The student may select the correct answer but this does not mean that the student understands the concept. Therefore the tutoring system must take into consideration the quality of an answer.

I developed a tutoring system which takes advantage of eye tracking technology and takes into consideration the effort given by the student. Beside the response from the student to a question, the tutoring system uses input from the eye tracking device
to create a model of the student’s reading pattern and to assess mental workload during question answers.

The reading tracker is an improved technique of measuring the student’s attention on screen. Several attempts [9,28] have been made before, but none of these techniques were able to tell the tutoring system if the student in fact read the presented concept because these techniques track only if the gaze position of the student falls within the boundaries of predefined rectangular regions. The technique of the reading tracker introduced in this thesis improves the model and provides more precise assessment of the student’s attention. This method is more precise, dynamic and does not require any input from the teacher to define the regions of the interest.

Next, I introduced a method for measuring effort while answering a question. This technique utilizes the ICA to determine if there is an increase in the mental workload. When answering questions, a student needs to apply what he/she learned before. Thinking about the answer or recalling facts from memory creates an increased mental workload which can be observed using the eye tracking device. The tutoring system has the ability to determine if there is an increased mental workload and use this new information to update the student knowledge state. The answer to a question, reading tracking data and mental workload assessment are seen by the system as the effort which the student gives in answering a question about a specific concept. These three inputs serve as the mechanism to determine how much effort is given by the student. This unique ability of the system creates a more precise model of the student’s knowledge state which leads to more efficient adaptation of the learning environment.

This tutoring system is also capable of dealing with the uncertainty which is often a big part of the student’s input. The student model is represented using the Bayesian Network to create a representation of the student’s knowledge state.

By separating the knowledge domain and assessment questionnaire from the tu-
toring system, it becomes easy to update the tutoring system with new content. This feature is beneficial to the teacher because the teacher does not need to have any knowledge about the tutoring system internal mechanics and he/she is still able to create or update the knowledge domain or assessment questionnaire.

The tutoring system offers adaptation of the learning environment using three different features. These features use the student model as input to tailor teaching instructions. The first form of adaptation of the system is Best Topic To Learn, which can find a concept in the knowledge domain which will be the easiest to understand, based on knowledge already learned by the student. Concept Prerequisites is the second feature of the system. This feature builds the pre-requisite list for each concept where each concept in the list must be first learned by the student in order to continue with the next concept. Lastly, the Learning Path Generator will generate a pre-requisite list for each concept in the selected goal and combine these lists into a learning path sequence. Concepts already learned by the student are omitted from the learning path sequence.

Because of the eye tracking device, the tutoring system presented here is implemented as a desktop application and therefore it is not possible to make this system accessible through the Internet. Currently, this system is deployed and operates in the Regina Integrative Cognitive Experimentation Lab, where the eye tracking device is located.

The tutoring system in this thesis offers an example and a reference on the application of eye tracking technology in the development of an intelligent tutoring systems. I believe that the tutoring system presented in this thesis is useful to any work applying eye tracking technology in the development of Intelligent Tutoring Systems.
6.2 Future Work

In the future, I plan to extend Intelligent Tutoring System measuring student effort using eye tracking technology to incorporate concepts of searching and sorting algorithms, such as bubble sort and binary search. Since the knowledge domain and Bayesian Network are implemented as separate modules from the tutoring system, extending the tutoring system to these new concepts will be efficient and require minimal new development.

Another feature I would like to implement in this tutoring system is the ability of the system to learn from collected data gathered during previous sessions. This ability will be then used to further adapt the tutoring system. This data could be used to identify questions which were most commonly answered incorrectly, or to identify questions which are important and delete questions which are no longer useful. In courses which are built to have a certain trail of continuity (i.e., Computer Programming subject could have multiple courses focusing on different programming languages), the data captured for one student could be used in subsequent courses for creating student’s initial knowledge state.

Another potential feature I would like to add to the system is a motivation module. Using the eye tracking device, the tutoring system is able to track both of the student’s eyes simultaneously. A motivation module will manage content presentation based on the student’s previous responses so the content is presented as a reward to the student. If the system detects that the student’s effort is decreasing, the motivation module will continue to present the content only if student closes one of his/her eyes. The student can have open both eyes only when he/she makes an effort to learn new concepts. If the student opens both eyes during a session where he/she can read only with one eye, the motivation module will automatically stop presenting content and remind the student to close one eye to continue. I speculate that discomfort created by reading content with one eye may motivate the student to learn better so he/she
can access additional content with both eyes open.

Currently this tutoring system is implemented as a desktop application with all data stored locally. I would like to extend the functionality of the system so it is available to access through a network from any device using cloud technology. New development in mobile technology brings opportunities for eye tracking to be part of a mobile device. Samsung’s Galaxy S4 and LG’s Optimus G Pro both will offer an eye-recognition feature that automatically reacts to the movement of the user’s eyes.

Lastly, I would like to add another communication channel between the student and the tutoring system. Electroencephalography (EEG) offers an opportunity to record voltage fluctuations within the brain. I would like to monitor the brain activity during a learning process at the point when the student is realizing a new knowledge and starts to understand new concepts. The signal produced by the EEG can be then analysed using the wavelet analysis method to discover if there is some pattern in brain activity when new knowledge is formed. If this pattern exists and can be detected, the tutoring system can use this information to observe when the student learns new concept. This could further improve the student model.
Bibliography


Appendix A

CODE SAMPLES

```c
void getLatestData(void)
{
    int err=check_recording();
    if(err!=0)
    {
        cout <<"\nERROR RECORDING DATA;"
    }
    else
    {
        "check for new sample update"
        if(eyelink_newest_float_sample(NULL)==)
        {
            eyelink_newest_float_sample(&evt);
            /* get gaze position from sample */
            eye_x = evt.fs.xg[eye_used];
            eye_y = evt.fs.yg[eye_used];
            /* make sure pupil is present*/
            if((eye_x!=MISSING_DATA && eye_y!=MISSING_DATA && evt.fs.ppa[eye_used]==0)
            {
                pupil_size(evt.fs.ppa[eye_used];
                /* add new data into container */
                eye_data_buffer.push_back(evt.fs.ppa[eye_used]);
            }
            else if((eye_x!=MISSING_DATA && eye_y!=MISSING_DATA)/"Blinks are not recorded for ICA computation*/
            {
                if(eye_data_buffer.back()>0.0f)
                eye_data_buffer.push_back(-0.0f);
            }
        }
    }
}
```

Figure A.1: Access to latest data produced by the eye tracking device.
void readWord(float x, float y)
{

    Word tempWord;
    for (std::list<Word>::iterator wordIterator = text.begin(), end = text.end(); wordIterator != end; ++wordIterator)
    {
        tempWord = *wordIterator;
        if (tempWord.flag1) {
            float x = (tempWord.x + tempWord.w) / 2.0f;
            float y = (tempWord.y + tempWord.h) / 2.0f;
            float a = abs(x - x);
            float b = abs(y - y);
            float c = sqrt((a * a) + (b * b));
            if (c <= BASE_WORD_DIST)
            {
                wordIterator->flag1 = true;
                break;
            }
        }
    }
}

Figure A.2: Method which updates each word as read.

void getPreRequisites(void)
{
    openList.clear();
    closedList.clear();
    prerequisites.clear();

    Page tempPage, tempPage2;
    float tempId;

    for (std::list<Page>::iterator pageIterator = pages.begin(), end = pages.end(); pageIterator != end; ++pageIterator)
    {
        tempPage = *pageIterator;
        getParent(getBinIndex(tempPage.pageId));

        while (!openList.empty())
        {
            tempId = openList.front();
            openList.pop_front();

            bool exist = false;
            for (std::list<float>::iterator it = closedList.begin(), end = closedList.end(); it != end; it++)
            {
                if (AlmostEqual(tempId, *it))
                {
                    exist = true;
                    break;
                }
            }

            bool exist2 = false;
            for (std::list<Page>::iterator pageIterator2 = pages.begin(), end = pages.end(); pageIterator2 != end; ++pageIterator2)
            {
                tempPage2 = *pageIterator2;
                if (AlmostEqual(tempId, tempPage2.pageId))
                {
                    exist2 = true;
                    break;
                }
            }

            if (!exist && !exist2)
            {
                closedList.push_back(tempId);
                getParent(getBinIndex(tempId));
            }
        }
    }
    prerequisites = (closedList);
    prerequisites.sort();
}

Figure A.3: Method which creates a list of pre-requisites.
void transform_signal( float* a, const int n )
{
    if (n >= 4) {
        int i, j;
        const int half = n >> 1;

        float* tmp = new float[n];

        for (i = 0, j = 0; j < n-3; j += 2, i++) {
            tmp[i] = a[j]^h0 + a[j+1]^h1 + a[j+2]^h2 + a[j+3]^h3;
            tmp[i+half] = a[j]^g0 + a[j+1]^g1 + a[j+2]^g2 + a[j+3]^g3;
        }

        tmp[i] = a[n-2]^h0 + a[n-1]^h1 + a[0]^h2 + a[1]^h3;
        tmp[i+half] = a[n-2]^g0 + a[n-1]^g1 + a[0]^g2 + a[1]^g3;

        for (i = 0; i < n; i++) {
            a[i] = tmp[i];
        }

        delete [] tmp;
    }
}

Figure A.4: Signal Transform Method.
```c
void openTrackerConnection(void)
{
    /* ... */
    if(open_eyeLink_connection(1))
        exit_eyeLink();
    set_offline_mode();
    flush_key_queue();
    eyeLink_ver = eyeLink_get_tracker_version(versstr);
    if (eyeLink_ver == 3)
        tracker_software_ver = get_tracker_sw_version(versstr);
    set_target_size(SCRWIDTH/88, SCRHEIGHT/330);
    set_calibration_colors(atten_text_color, &background_color);
    set_cal_sounds("off", "off", "off");
    set_door_sounds("off", "off", "off");
    if(tracker_file_name[0])
    {
        if(!strstr(tracker_file_name, ",",)) 
            status(tracker_file_name, ".EDF");
        int result = open_data_file(tracker_file_name);
        if(result==0)
        {
            alert_printf("Cannot create EDF file 'ks', tracker_file_name");
            exit_eyeLink();
        }
        eyedc_printf("add_file_preamble_text 'RECORDED BY Ks', "ITS with Eyelink II program");
    }
    eyedc_printf("screen_pixel_coords = %ld %ld %ld %ld", dispInfo.left, dispInfo.top, dispInfo.right, dispInfo.bottom);
    eyedc_printf("calibration_type = %s",);
    eyedc_printf("DISPLAY COORDS %ld %ld %ld %ld", dispInfo.left, dispInfo.top, dispInfo.right, dispInfo.bottom);
    if(dispInfo.refresh>40)
        eyedc_printf("FREQUATE %1.2f Hz ", dispInfo.refresh);
    if(eyeLink_ver==2)
    {
        eyedc_printf("select_parser_configuration 0");
        if(eyeLinkVer == 2)
        {
            eyedc_printf("scene_camrage_gazeMap = NO");
        }
    }
    else
    {
        eyedc_printf("saccade_velocity_threshold = 35");
        eyedc_printf("saccade_acceleration_threshold = 9500");
    }
}
```

Figure A.5: Method to open and setup connection to the eye tracker.
Appendix B

REB APPROVAL FORM
DATE: February 4, 2013

TO: Peter Lach
     841 Gladmer Park
     Regina, Sk. S4P 2X7

FROM: Dr. Larena Hoeber
      Chair, Research Ethics Board

Re: Intelligent Tutoring Systems Measuring Students Effort Using Eyetracking Technology
    (File # 47S1213)

Please be advised that the University of Regina Research Ethics Board has reviewed your proposal and found it to be:

☐ 1. APPROVED AS SUBMITTED. Only applicants with this designation have ethical approval to proceed with their research as described in their applications. For research lasting more than one year (Section 1F). ETHICAL APPROVAL MUST BE RENEWED BY SUBMITTING A BRIEF STATUS REPORT EVERY TWELVE MONTHS. Approval will be revoked unless a satisfactory status report is received. Any substantive changes in methodology or instrumentation must also be approved prior to their implementation.

☐ 2. ACCEPTABLE SUBJECT TO MINOR CHANGES AND PRECAUTIONS (SEE ATTACHED). Changes must be submitted to the REB and approved prior to beginning research. Please submit a supplementary memo addressing the concerns to the Chair of the REB. ** Do not submit a new application. Once changes are deemed acceptable, ethical approval will be granted.

☐ 3. ACCEPTABLE SUBJECT TO CHANGES AND PRECAUTIONS (SEE ATTACHED). Changes must be submitted to the REB and approved prior to beginning research. Please submit a supplementary memo addressing the concerns to the Chair of the REB. ** Do not submit a new application. Once changes are deemed acceptable, ethical approval will be granted.

☐ 4. UNACCEPTABLE AS SUBMITTED. The proposal requires substantial additions or redesign. Please contact the Chair of the REB for advice on how the project proposal might be revised.

[Signature]
Dr. Larena Hoeber

cc: Dr. Cory Butz – Computer Science
    Dr. Brien Maguire – Computer Science

** supplementary memo should be forwarded to the Chair of the Research Ethics Board at the Office for Research, Innovation and Partnership (Research and Innovation Centre, Room 109) or by e-mail to research.ethics@uregina.ca